ACEEE Int. J. on Information Technology, Vol. 3, No. 1, March 2013

An Area Efficient Vedic-Wallace based Variable Precision Hardware Multiplier Algorithm

Neelima Koppala[1], Rohit Sreerama[2], and Satish Paidi [2]

[1] Assistant Professor, ECE Department, Sree Vidyanikethan Engineering College (Autonomous), A.Rangampet, Tirupati, India-517102.

[2] M.Tech (VLSI), ECE Department, Sree Vidyanikethan Engineering College (Autonomous), A.Rangampet, Tirupati, India-517102.

Email: [1] koppalaneelima@gmail.com,{rohit.sreerama, paidysatish} @gmail.com

***Abstract*—The complete architecture with the necessary blocks and their internal structures are proposed in this paper. In this algorithm the complete variable precision format is utilized for the multiplication of the two numbers with a size of nxn bits. The internal multiplier is choosen for m bit size and is implemented using vedic-wallace structure for high speed implementation. The architecture includes the calculation of all the fields in the format for complete output. The exponent of the final result is obtained by using carry save adder for fast computations with less area utilization. This multiplier uses the concept of MAC unit, giving rise to more accurate results having a bits size of the final result will be $2n^2$.**

***Index Terms*—Variable Precision, MAC, Vedic-Wallace Structure, Carry Save Adder.**

I. Introduction

With the advent and improvements in VLSI technology that has enabled the integration of transistors based on different technologies like CMOS,BiCMOS technologies, etc., the speed of computation has increased dramatically. To obtain the performance, parallel processing and pipelining techniques are used to perform various operations, which is obtained with the tradeoff with accuracy. Hence accuracy is not developed in proportion with speed when large operands are considered. Due to errors like accumulation of rounding errors and catastrophic cancellation in the systems used, the results obtained were completely inaccurate causing costly destruction such as Ariane5 rocket during first flight in 1996 or life destruction such as the overdoses of radiation given to patients in 2000 at National Oncology Institute of Panama [3].

As arithmetic operations are very important in the design of digital processors and application specific circuits. Arithmetic circuits form an important class of circuits in digital systems. As the numbers are represented as real numbers in computers, several ways to represent them were developed. Among those, Floating-point representation, in particular the standard IEEE format [21], is by far the most common way of representing an approximation to real numbers in computers because it is efficiently handled in most large computer processors and can support a much wider range of values. Binary fixed point is usually used in special-purpose applications on embedded processors that can only do integer arithmetic, but decimal fixed point is common in commercial applications. Fixed point places a radix point somewhere in the middle of the digits, and is equivalent to using integers that represent portions of some unit. Fixed-point has a fixed window of representation, which limits it from representing very large or very small numbers. Also, fixed-point is prone to a loss of precision when two large numbers are divided. Floating-point solves a number of representation problems [2]. Floating-point employs a sort of "sliding window" of precision appropriate to the scale of the number. This allows it to represent numbers from 1,000,000,000,000 to 0.0000000000000001 with ease. The term floating point refers to the fact that the radix point (decimal point, or, more commonly in computers, binary point) can "float"; that is, it can be placed anywhere relative to the significant digits of the number. This position is indicated separately in the internal representation, and floating-point representation can thus be thought of as a computer realization of scientific notation.

Multiplication is the scaling operation performed one number by another used frequently in various applications like convolution, Fast Fourier Transforms, filters, ALU of Processors, etc. Hence the development of fast and accurate multipliers is necessary based on the precision considered. This paper presents the complete architecture and its details for variable precision.

II. Variable Precision Format Representation

According to the IEEE standards, particular formats are devised for various representations. When the communication between processors is considered, if the significant bits can be accommodated with more number of bits then the precision increases which yield highly accurate results. Variable precision floating point format, gives the designer a choice to accommodate large number of bits as required for the design. The representation of variable precision format [1] is shown in fig.1, where Exponent Field (E) can have extended bits up to 16. The Sign Field (S) is considered to have single bit that is assigned with 1 if the signed operand is positive and 0 if two bits which indicate whether the operands are normalized, infinite, zero and NaN to take care of exceptional cases. The Length Field (L) has five bits which signify the number of m bit variables present in the operand. The Significand Field (F) has L+1 i.e., F(0) to F(L) locations each capable of storing specified m bits of the

DOI: 01.IJIT.3.1.1113

ACEEE Int. J. on Information Technology, Vol. 3, No. 1, March 2013

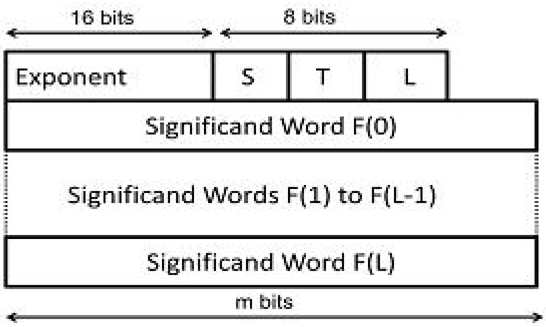

Fig. 1. Representation of Variable Precision format

operands. The value of a variable precision number is specified by VP=$(-1)^s$xFx2^E , which have a maximum precision of 32m bits whose range is approximately $[2^{-32768},2^{32769}]$H”$[10^{-9864}, 10^{9865}]$.

III. Architecture Of Variable Precision Multiplier

The architecture concentrates mainly on variable precision multiplication according to the variable precision format [3]. The result of the multiplication depends on the operands considered. The normalized operands are used for multiplication of significant parts. The output sign bit is calculated from the input sign bits by performing xor operation. So that if any one of the operand is negative then the resultant operand is negative as shown in fig.2.

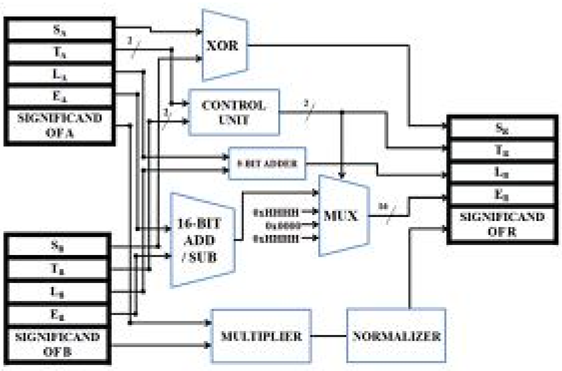

Fig. 2. Proposed architecture for variable precision multiplier

The control unit decides the resultant type of the operand which includes the exceptional cases of the operation result as shown in Table I .The operation to be performed by the control unit is based on the obtained result as shown in Table II.

The adder circuits [6] are considered for operation are carry save adder so that the critical path delay is less. The multiplier is considered for the significant multiplication is based on the variable precision algorithm [1] as shown in fig.3.

Variable Precision Multiplication Algorithm

Inputs A, B
Input m
Output R= A×B
Initialization R= 0
For j=0 to n-1 do
For i=0 to n-1 do
R=R+ $(A_i \times B_j) \times 2^{(m(i+j))}$
End for
End for
Return R = $(R_0, R_1, \ldots, R_{2n-1})$

Fig. 3. Variable Precision Algorithm

Table I. Result Of Control Unit

Operand A -type	Operand B-type	Result
Normalized	Normalized	Normalized
Normalized	Infinite	Infinite
Normalized	Zero	Zero
Normalized	NaN	NaN
Infinite	Normalized	Infinite
Infinite	Infinite	NaN
Infinite	Zero	NaN
Infinite	NaN	NaN
Zero	Normalized	Zero
Zero	Infinite	NaN
Zero	Zero	Zero
Zero	NaN	NaN
NaN	Normalized	NaN
NaN	Infinite	NaN
NaN	Zero	NaN
NaN	NaN	NaN

Table II. Control Unit Operation Table

Result	Operation to be done	S-field	E-field
Normalized	Multiply	xor	Add/Sub
Infinite	Assign infinity	xor	0XHHHH
Zero	Assign zero	xor	0X0000
NaN	Assign NaN	Don't care	0XHHHH

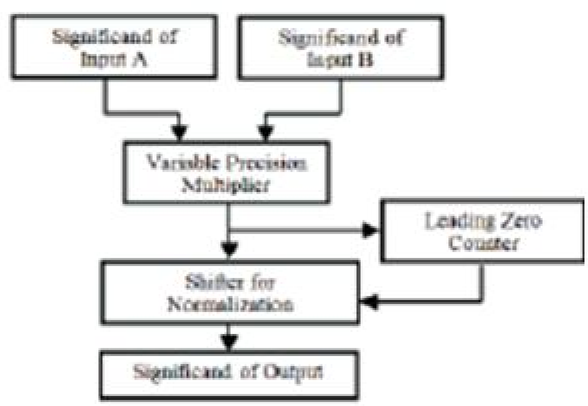

Fig. 4. Block diagram for multiplication of significand inputs.

One of the main operations of floating point data path is normalization. It involves the use of a leading zero counting (LZC) [2] to correctly update the exponent part of the result, or a detection unit and a normalization shifter that shifts the result to left according to the outcome of the LZC unit. The shifter used is the barrel shifter to shift the smaller number to the right by the difference in one cycle as shown in fig.4.

IV. Vedic Wallace Multiplier

In general Wallace tree addition uses full adders to

DOI: 01.IJIT.3.1.1113

extensively reduce the partial products. When the critical path is compared between critical path in 4 bit conventional and Vedic multiplier, for a 4 bit multiplier, 4 partial products will be generated, as shown in fig.5 and are named as P0 to P3. For Wallace tree multiplier a 3:2 reduction is used, so that the partial products are reduced from 4 to 3. The Delay in critical path is given by addition of 3 full adder sums, 2 full adder carry, and half adder carry.

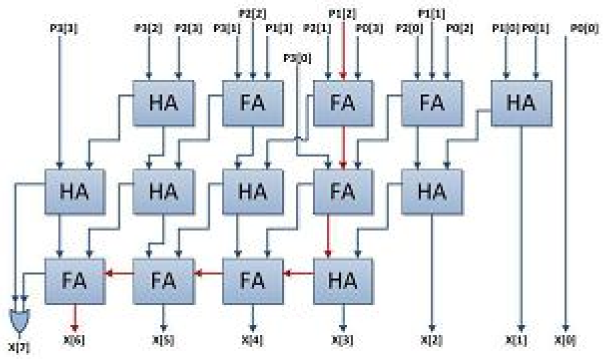

Fig. 5. 4x4 Multiplier using Wallace Tree

The critical path for the Vedic[4] [5] mathematics as shown in figure.6, the critical path is given by 2 FAS is reduced by 3HASand in terms of XOR gates Vedic-Wallace uses 3XOR gates instead of 4XOR i.e., less carry propagation delay than conventional method. Hence Vedic-Wallace has a variable improvement over design ware depending upon the number of bits in multiplication

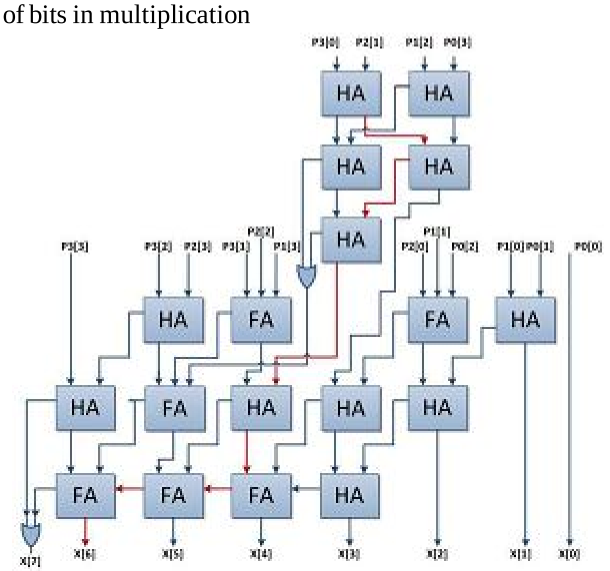

Fig. 6. 4x4 Multiplier using Vedic Reduction

V. Implementation Results

The Complete Architecture is implemented on Xilinx ISE10.1i Tool where the design is mapped to XC3s500efg320-5 FPGA device with a speed grade of -5. The following table shows the simulation and synthesized results of the design using Modified Booth Multiplier [3] and Vedic-Wallace Multiplier.The synthesized results show that even though the number of slices utilized are same and the combinational path delay is more for Vedic-Wallace multiplier, it proves

Table III. Synthesized Device Mapped Results

Basic multiplier considered	No.of Slices	No.of 4-input LUTs	No .of IOs	No. of bonded IOBs	No. of MULT18X18SIOs	Maximum Combinational Path Delay
Modified Booth Multiplier	416 out of 4656 8%	769 out of 9312 8%	200	200 out of 232 86%	20 out of 20 100%	54.876ns
Vedic-Wallace Multiplier	375 out of 4656 8%	700 out of 9312 7%	200	164 out of 232 70%	--	62.411ns

efficient as none of built in multipliers are used and the number of LUTs are also less proving the design to be area efficient.

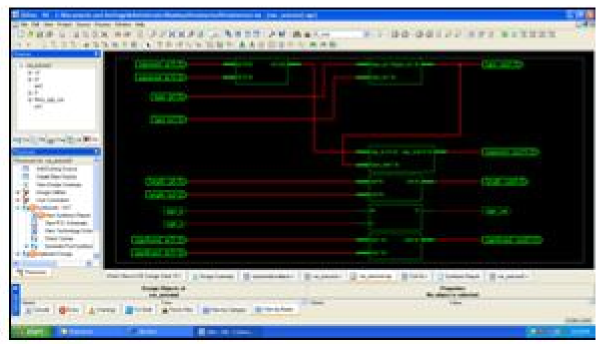

Fig. 7. The synthesized RTL Schematic

The fig.7 shows the developed RTL Schematic which shows the complete details of designed architecture with the required fields like sign, type, length, exponent, significant, etc. Fig.8 shows the simulated output for verifying the accuracy of the design. The value of exponent is 16’hFFFF when the sign bit of input ‘a’ raises to ‘1’ which signifies the number to be a negative number and its type to be “01” which indicates that the output is infinite. The significant values are obtained to be accurate where for sample input a=32’h00000003 and b=32’h00000032 yielding an output of 32’h00000096.

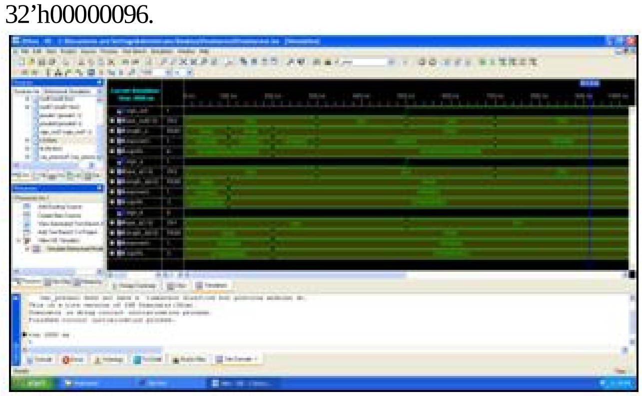

Fig. 8. The Simulated output for both designs

Conclusions

This paper concludes that the complete architecture in the

DOI: 01.IJIT.3.1.1113

ACEEE

variable precision format is developed and verified using Xilinx ISE10.1i Tool with the design mapped to the XC3s500 FPGA with 320 IO pins and a speed of -5 Grade. The design gives accurate results for any desired number represented in variable precision format. The implementation of basic multiplier was considered for Modified Booth Multiplier and Vedic-Wallace Multiplier and Vedic-Wallace was found to be area efficient than Modified Booth Multiplier. Hence for generic applications where area is a constraint this design proves to be efficient for large prototyping devices.

References

[1] Rohit Sreerama, Paidi Satish, K Neelima. "An Algorithm for variable precision based floating point multiplication", *proc* International Conference on Advances in Information Technology and Mobile Communication, AIM 2012, page no-238-242.

[2] Satsh Paidi, Rohit Sreerama, K.Neelima "A Novel High Speed Leading Zero Counter for Floating Point Units" International Journal of Engineering Research and Applications(IJERA), Vol.2, Issue 2, Mar-Apr 2012, pp.1103-1105.

[3] Neelima Koppala, Rohit Sreerama and Paidi Satish. " Performance Comparison of Fast Multipliers Implemented on Variable Precision Floating Point Multiplication Algorithm" International Journal of Computer Applications in Engineering Sciences, Vol-II, Issue-II, June 2012, Pages: 56-59.

[4] Sumit R. Vaidya, D. R. Dandekar "Performance Comparison of Multipliers for Power-Speed Trade-off in VLSI Design" Recent Advances In Networking, Vlsi And Signal Processing.pg 263-266.

[5] Pushpangadan R, Sukumaran V, Innocent R, Sasikumar D, Sundar V. "High Speed Vedic Multiplier for Digital System Processors. IETE J Res [serial online] 2009:55:282-6.

[6] Raghunath, R.K.J.; Farrokh, H.;Naganathan, N.; Rambaud, M.; Mondal, K.; Masci, F. Hollopeter "A compact carry-Save multiplier architecture and its applications" Circuits and Systems, 1997. Proceedings of the 40th Midwest Symposium Page(s): 794 - 797 vol.2.

DOI: 01.IJIT.3.1.1113